explicit teachings of the reference. As demonstrated below, there is ***no rational basis*** for the Examiner's assertions that Hyndman et al. discloses each and every limitation in the claim, let alone each and every element as arranged in the claim.

It is well settled that "[t]he identical invention must be shown in as complete detail as is contained in the ... claim." Richardson v. Suzuki Motor Co., 868 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). "Anticipation cannot be predicated on teachings in the reference which are vague or based on conjecture." Studiengesellschaft Kohle mbH v. Dart Industries, Inc., 549 F. Supp. 716, 216 USPQ 381 (D. Del. 1982), aff'd., 726 F.2d 724, 220 USPQ 841 (Fed. Cir. 1984).

Further, anticipation cannot be established based on a piecemeal application of the reference, where the Examiner picks and chooses isolated features of the reference in an attempt to synthesize the claimed invention. "Anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim." *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984). Hence, it is not sufficient that a single prior art reference discloses each element that is claimed, but the reference also must disclose that the elements are arranged as in the claims under review. *In re Bond*, 15 USPQ2d 1566, 1567 (Fed. Cir. 1990) (citing *Lindemann Maschinenfabrik GmbH*).

The Examiner provides a tortured interpretation of the reference in an attempt to synthesize features that are not actually disclosed in the reference. For example, the Examiner asserts on page 9 (paragraph a. of Response to Arguments) that "Hyndman teaches the movement of *objects* [sic] and data throughout the network, *which constitutes the movement of selected objects between nodes* (column 2, lines 32-40; column 5, lines 56-64; column 6, lines 34-39).

A review of the cited portions, however, demonstrates that Hyndman does not teach the movement of "objects" as claimed. In particular, col. 2, lines 32-40 states:

> According to one aspect of the invention there is provided a user interface for a network management system of a communication network, comprising a server for storing the state of the user interface as a whole, marshalling data, and interacting with the remaining of the network management system, a client layer for using the server for refreshing a current display presentation of network data to a user, and for transmitting commands selected by a user on the current display presentation to the server.

This cited portion of col. 2, lines 32-40, however, does not disclose or suggest the claimed ***transferring a selected service object*** (including any one of a model object, a view object and a controller object), where the service object is able to provide a first network service based on exchange of service transaction messages with other associated service objects. In other words, the claimed ***service object*** is the entity that exchanges service transaction messages, and it is the ***service object*** that is transferred between nodes.

In contrast, the cited portion of Hyndman simply describes "marshalling data" and "transmitting commands", with no reference whatsoever to transferring a ***selected service object*** (e.g., a model object, a view object, or a controller object) that provides service based on exchange of service transaction messages. As described below, Hyndman at best describes only transfer of messages (e.g., event messages), *not* service objects.

Column 5, lines 56-64 of Hyndman states:

> Providers can define custom commands which are loaded from the CCBB 34 (Custom Command [Building Block]) at UIS [user interface server] start up. The custom command controllers are responsible for catching custom command events, parsing the URL for parameters and replacing them with data stored in the UIS. Finally, the custom command controller will generate a display URL event which will be caught by the view responsible for displaying the URL at the UIC [user interface client].

The Examiner's assertion that this portion of Hyndman "teaches the movement of objects and data throughout the network which *constitutes the movement of selected objects between nodes*" is nonsensical at best. In fact, the Examiner deliberately ignores the architecture of Hyndman, which demonstrate that the above-quoted portion of Hyndman simply describes configuring the user interface server for presentation of selected display data to the user interface client.

As illustrated in Figures 1B and 2, the architecture of Hyndman utilizes a User Interface Server (UIS) 12 executed on a web server and that implements the model object M-2 of Fig. 1B (col. 6, lines 9-12 and 23-26). In particular, the fundamental teaching of Hyndman is that the user interface is separated into a server layer and a client layer, and according to a multilevel model-view-controller pattern:

> An important change from the prior art architectures resides in the separation of the user interface (UI) into a server layer and a client layer user interface, namely the user interface server (UIS) and the user interface client (UIC).
>
> Another key feature of the architecture used for the user interface according to the invention is that it follows a multilevel model-view-controller (MMVC) pattern, rather than a MVC pattern. As a result, the amount or communication required between the client (the view), the server (the controller) and database/network (the model). [sic] MMVC is a nested MVC [sic] is considerably reduced [sic, presumably intended to state that the required communication is reduced]. ***At each level a view contains another MVC pattern***. MMVC also supports multiple views on the same data.
>
> (Col. 3, lines 44-57).

Hence, Hyndman stresses that the important feature is that the multilevel MVC pattern provides another MVC pattern within each view. There is no teaching or suggestion of ***transferring*** a ***selected service object***, as claimed, but rather the teaching is partitioning a view object into another collection of model-view-controller objects.

This teaching of partitioning a view object is further described by the description of the User Interface Client and the User Interface Server. In particular, Figure 1B illustrates that the User Interface Client (UIC) (11 of Fig. 2) is implemented, according to a model-view-controller (MVC) architecture, by implementing model M-1 (a cache for data stored in the model M-2), controller C-1, and a view V-1 within a web browser of a user terminal (col. 2, lines 26-31; col. 6, lines 11-12; col. 7, lines 44-45). The UIC (11 of Fig. 2) having the objects M-1, C-1, and V-1 **also forms the view V-2** (col. 4, lines 1-16). The UIC 11 "uses the services of the server layer and is responsible for presentation of network data to the user and for general interaction with the user. UIC 11 manages only enough data to refresh the current display and to prevent excessive messaging." (Col. 5, lines 36-40).

The User Interface Server (UIS) (12 of Fig. 2) is executed within a host computer executing a web server (col. 6, lines 8-10) and is implemented as the model M-2 of Fig. 1B (col. 4, lines 5-9; col. 6, lines 23-26). The **controller C-2 is the data marshalling mechanism** (col. 4, lines 7-9) on the server (col. 4, lines 18-19). The UIS of the server layer "comprises a collection of building block

controllers which interact with the [building blocks Bbs] 3, for storing the state of the user interface as a whole, marshalling data, and interacting with the remaining of the network management system." (Col. 5, lines 29-35).

Hence, "the model of the UIS (M-2 in FIG. 1B) keeps the state of all objects the user has access to, and propagates any state changes that occurs to objects currently being displayed. The state of an object can be affected by a message from a BB or by a user command." (Col. 6, lines 23-27).

Consequently, all interactions between the model M-2 and the view V-2 (i.e., the UIC executing the model M-1, view V-1, and controller C-1) are limited to: the model M-2 receiving commands from the view V-2 (see Figs. 4A and 4B and col. 6, lines 47-52, especially step 52), and the view V-2 receiving event updates from the model M-2. (See, e.g., Figs. 3, 5A and 6 and col. 6, lines 39-46, col. 7, lines 7-19 and 46-53).

Hence, the reference to col. 6, lines 34-39 simply refers to an acylic direct graph (i.e., a directed acyclic graph) that enables event data to be propagated toward the UIC 11: if necessary, the even data may be intercepted by the UIS (M-2) (see col. 6, lines 23-33).

Hence, the recital at column 6, lines 34-39:

> The graph event propagation mechanism allows events to travel to nodes up or down the graph hierarchy. Certain types of events that are received at a node may be allowed to continue to propagate through the hierarchy, other types may be intercepted. This graph event propagation mechanism also allows filtering and collation of graph events.

simply describes the propagation of event messages toward the directed acyclic graph toward the UIC 11.

Consequently, Hyndman requires that **all the model-view-objects are *fixed* in their respective positions**. There is absolutely no disclosure whatsoever of the claimed "***transferring*** a ***selected service object*** between any one of the service node, the network-enabled user interface device, and a second network node" as claimed. Rather, Hyndman simply describes the conventional transfer of event messages or view commands between the User Interface Client (i.e., V-2) and the User Interface Server (i.e., M-2). Any assertion that an event message or a view command is in any way equivalent to the claimed service object is without foundation in fact or law, and is per se

unreasonable.

Hence, the §102 rejection ***must*** be withdrawn because it fails to demonstrate that the applied reference discloses each and every element of the claim.

Claims 1-3 and 7-9 stand rejected under 35 USC 103 (a) in view of Hyndman et al. and US Patent No. 5,926,177 to Hatanaka et al. This rejection is respectfully traversed.

As described above with respect to claims 13, 20, and 27, independent claims 1 and 7 specify an arrangement in a network-enabled user interface device, where at least one service object (e.g., a model object, a view object, or a controller object) for a first network service is ***received via the open protocol network***, and the network-enabled user interface device executes the at least one service object for the first network service.

As such, the above arguments describing Hyndman et al are incorporated in their entirety herein by reference: there is no disclosure or suggestion whatsoever of ***receiving*** at least one ***service object*** via an open protocol network, as claimed.

Further, Hatanaka et al. provides no disclosure whatsoever for the claimed selective termination of the received one service object (providing the first network service) ***based on reception, via the open protocol network, of a second service object,*** as claimed.

Rather, Hatanaka et al. simply teaches that old display data is closed as new graphical information is presented. The portion cited by the Examiner stresses that the View object is replaced, and not necessarily ***terminated.*** Further, the view object is replaced not in response to the claimed ***reception, via the open protocol network, of a second service object***, but rather in response to the user manually selecting a change in the view:

> The second approach has been to replace the View object in the Model-View-Controller configuration **as the type of information the user is interested in changes**. **When the user is no longer interested in the structure** of the distributed application and **becomes interested in the detailed state of the components**, **the structure view is replaced with a detailed state view**. The problem with this approach is that the new view must be recreated on demand and inserted into the Model-View-Controller configuration. ***Even if the old view is preserved*** *when the user switches out of a particular mode (switching from structure to state views, for example), there is expense involved when the old view is reused (switching back to structure from state). The old view must be reconnected with the model and controller, and it must be made current to reflect any model*

changes that occurred when it was not active. This expense is enough to inhibit the user from switching between views freely.

Hence, Hatanaka et al. does not even teach ***terminating*** the one service object, but rather **switching** the view object, and even suggests preserving the old view object while not in use! By definition the preservation of the old view object precludes terminating the old view object.

Consequently, the hypothetical combination of Hyndman et al. and Hatanaka et al. neither discloses nor suggests, singly or in combination, ***receiving*** first and second service objects ***for the respective first and second network services***, where one service object ***for the first network service*** is selectively terminated based on reception, via the network interface, of the second service object for the corresponding second network service. Hence, the hypothetical combination neither discloses nor suggests: (1) receiving the one and second service objects via the open protocol network; or (2) selectively ***terminating*** the one service object based on reception of the second service object for a corresponding second network service.

An evaluation of obviousness must be undertaken from the perspective of one of ordinary skill in the art addressing the same problems addressed by the applicant in arriving at the claimed invention. Bausch & Lomb, Inc. v. Barnes-Hind/Hydrocurve, 23 USPQ 416, 420 (Fed. Cir. 1986), cert. denied, 484 US 823 (1987). Thus, the claimed structures and methods cannot be divorced from the problems addressed by the inventor and the benefits resulting from the claimed invention. In re Newell, 13 USPQ2d 1248, 1250 (Fed. Cir. 1989).

At best, the hypothetical combination simply uses the same view object to manage multiple views in respective windows, where only active windows will display changes. Hence, there is no disclosure or suggestion in the hypothetical combination of dynamically providing users with ***different network services*** based on reception of associated service objects via the open protocol network.

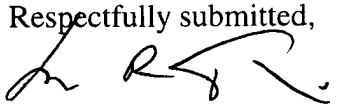
For these and other reasons, the §103 rejection should be withdrawn.

The indication of allowable subject matter is claims 4-6, 10-12, 15-18, 22-24, and 29-31 is acknowledged with appreciation.

In view of the above, it is believed this application is and condition for allowance, and such a Notice is respectfully solicited.

To the extent necessary, Applicant petitions for an extension of time under 37 C.F.R. 1.136. Please charge any shortage in fees due in connection with the filing of this paper, including any missing or insufficient fees under 37 C.F.R. 1.17(a), to Deposit Account No. 50-1130, under Order No. 95-468, and please credit any excess fees to such deposit account.

Respectfully submitted,



Leon R. Turkevich
Registration No. 34,035

Customer No. 23164
(202) 261-1059
Date: October 19, 2005